| L Number | Hits | Search Text | DB | Time stamp |
|---|---|---|---|---|
| 1 | 86 | duplicat$3 adj backup$3 | USPAT; US-PGPUB | 2002/11/01 15:44 |
| 4 | 74 | duplicat$3 adj backup$3 | USPAT | 2002/11/01 15:44 |
| 5 | 5 | 4.ab,clm,ti. | USPAT | 2002/11/01 15:46 |
| 6 | 0 | (portion$1 or partial$3) near2 backup43 | USPAT | 2002/11/01 15:47 |
| 7 | 0 | (portion$1 or partial$3) near5 backup43 | USPAT | 2002/11/01 15:47 |
| 8 | 867 | (portion$1 or partial$3) near2 backup$3 | USPAT | 2002/11/01 15:50 |
| 9 | 1 | ((portion$1 or partial$3) near2 backup$3) same (duplicat$3 adj backup$3) | USPAT | 2002/11/01 15:50 |
| 10 | 10 | ((portion$1 or partial$3) near2 backup$3) and (duplicat$3 adj backup$3) | USPAT | 2002/11/01 15:50 |
| 11 | 10 | delta near2 backup$3 | USPAT | 2002/11/01 15:55 |
| 12 | 22 | (backup$3 adj data) same ((portion$1 or partial$3) near2 backup$3) | USPAT | 2002/11/01 16:03 |
| 13 | 5 | (backup$3 adj data) near2 again | USPAT | 2002/11/01 16:05 |
| 14 | 1 | (backup$3 adj data) near5 ((two or second$4) adj level) | USPAT | 2002/11/01 16:22 |
| 15 | 1 | ((backup$3 adj data) same ((portion$1 or partial$3) near2 backup$3)) and ((backup$3 adj data) near5 ((two or second$4) adj level)) | USPAT | 2002/11/01 16:22 |

| | ~~Document ID~~ | Title | Current OR |
|---|---|---|---|
| 1 | US 5778395 A | System for backing up files from disk volumes on multiple nodes of a computer network | 707/204 |

US-PAT-NO: 5778395
DOCUMENT-IDENTIFIER: US 5778395 A

TITLE: System for backing up files from disk volumes on multiple nodes of a computer network

DATE-ISSUED: July 7, 1998

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Whiting; Douglas L. | Carlsbad | CA | N/A | N/A |
| Dilatush; Tom | Chula Vista | CA | N/A | N/A |

US-CL-CURRENT: 707/204; 707/10

ABSTRACT:

A system for backing up files from disk volumes on multiple nodes of a computer network to a common random-access backup storage means. As part of the backup process, duplicate files (or portions of files) may be identified across nodes, so that only a single copy of the contents of the duplicate files (or **portions thereof) is stored in the backup** storage means. For each backup operation after the initial backup on a particular volume, only those files which have changed since the previous backup are actually read from the volume and stored on the backup storage means. In addition, differences between a file and its version in the previous backup may be computed so that only the changes to the file need to be written on the backup storage means. All of these enhancements significantly reduce both the amount of storage and the amount of network bandwidth required for performing the backup. Even when the **backup data** is stored on a shared-file server, data privacy can be maintained by encrypting each file using a key generated from a fingerprint of the file contents, so that only users who have a copy of the file are able to produce the encryption key and access the file contents. To view or restore files from a backup, a user may mount the backup set as a disk volume with a directory structure identical to that of the entire original disk volume at the time of the backup.

24 Claims, 14 Drawing figures

Exemplary Claim Number: 1

Number of Drawing Sheets: 11

---------- KWIC ---------

Abstract Text - ABTX:

A system for backing up files from disk volumes on multiple nodes of a computer network to a common random-access backup storage means. As part of the backup process, duplicate files (or portions of files) may be identified across nodes, so that only a single copy of the contents of the duplicate files (or **portions thereof) is stored in the backup** storage means. For each backup operation after the initial backup on a particular volume, only those files which have changed since the previous backup are actually read from the volume and stored on the backup storage means. In addition, differences between a file and its version in the previous backup may be computed so that only the changes to the file need to be written on the backup storage means. All of these enhancements

significantly reduce both the amount of storage and the amount of network bandwidth required for performing the backup. Even when the **backup data** is stored on a shared-file server, data privacy can be maintained by encrypting each file using a key generated from a fingerprint of the file contents, so that only users who have a copy of the file are able to produce the encryption key and access the file contents. To view or restore files from a backup, a user may mount the backup set as a disk volume with a directory structure identical to that of the entire original disk volume at the time of the backup.

Detailed Description Text - DETX:

In yet another embodiment, the backup storage means incorporates hierarchical storage management (HSM), in which files that have not been accessed for a long time are migrated from disk to a secondary storage means, such as tape or optical disk. The main purpose of HSM is to save on storage costs for very large storage systems by providing the management tools that allow the migration to be transparent to the system, except for the additional delay in accessing some files. Use of any form of HSM in conjunction with the backup storage means of the present invention does not significantly affect any of the concepts discussed here. However, care must be taken not to impair performance of the backup and restore operations, since delays incurred in accessing secondary storage may render the system much less usable. Indeed, it would be fairly simple to identify **portions of the contents of backup data** and directory files of the present invention which could be migrated to secondary storage without adversely affecting backup performance. Fortunately, in most cases, the data reduction methods of the present invention are sufficiently powerful to keep disk storage costs down to an acceptable level even without using HSM.

Detailed Description Text - DETX:

The backup process of the preferred embodiment actually creates two files containing information about each backup set: a backup directory file (e.g., 143), and a **backup data** file (e.g., 144). In an alternate embodiment, these files could be combined into a single file. The contents of the backup directory file indicate the directory structure of the source disk volume, as well as pointers into the **backup data** files (e.g., 144, 149, and **backup data** files of other users) indicating where the data for each file is to be found. One key feature of the present invention is the data reduction achieved by duplicating pointers to data and directory information instead of duplicating the information itself, including referencing duplicate information across users. To explain the role of the backup directory file(s) in accomplishing this data reduction in the preferred embodiment, a description of key **portions of the backup** directory file (e.g., 143) for a DOS disk volume is given in FIG. 4 in Backus-Naur Form (BNF), which is a well known formal language technique (for example, see Nicklaus Wirth, Algorithms+Data Structures=Programs, 1976, pp. 281-291). Before discussing the contents of FIG. 4, we will explicitly define the conventions of our BNF, since there are slight variations in syntax from one author to the next. Non-terminals are enclosed in angle brackets (e.g., <fileEntry>) . The :==symbol indicates a formal definition. Terminals are indicated as single binary digits (0 or 1), or as hexadecimal quantities using C-like syntax: 0xUU for 8-bit bytes, 0xUUUU for 16-bit words, and 0xUUUUUU for 32-bit dwords. Ranges of terminal values are indicated as two terminal quantities with two periods in between; e.g., 0.times.00 . . . 0.times.FE. The .vertline. character is a meta-symbol indicating "one or the other", while brackets [ ] indicate an optional field, and an asterisk (*) indicates one or more repetitions of the field. Thus, for example, [<externDirItem>]* indicates zero or more of the non-terminal <externDirItem>. The double slash // indicates a comment to the end of the line.

Detailed Description Text - DETX:

With the counts array 502 used to represent the first m.sub.0 bits of the <dirInfoCRC> value 439 very efficiently, the remainder of the first level 500 consists of an array 505 of N entries, packed at a bit level. Each entry contains x bits of the <dirInfoCRC> value 439 (beyond the most significant m.sub.0 bits) and y bits of the <partialFileCRC> 440. The values x and y are chosen by the Agent 108 (and stored in the global database file header) based on mo and the total number of entries N in the global database 145. Since the entire first level 500 is downloaded, the idea is to trade off the size of the array 505 to minimize the number of accesses required into the second level 502 to validate a match. For example, using N and M from the above example, if we choose x=10 bits and y=0 bits, then the table 505 consists of a total of about 1220K bytes (one million entries at 10 bits each); the entire first level 500 consists of about 1260 K bytes (1220K+40K), as opposed to the 16M bytes required for a complete download of the entire database. Since we have m.sub.0 +x=26 bits of <dirInfoCRC> 439 thus represented by the first level 500, the average probability pf of a false second-level match based on a first-level match is then given roughly by pf =N/2.sub.26 .about.1/64, assuming (as we are) a random distribution of <dirInfoCRC> values 439 in the database. In other words, when filtering database inquiries at the first level, about 63 of 64 inquiries that match at the first level will result in matches at the second level also, in this example. Since every inquiry into the second level 501 involves a disk access into an entry (e.g., 513) containing the remaining fields of the global database entry, it is important to minimize spurious accesses. Typically a value of p.sub.f in the range 1/16 to 1/256 gives a reasonable tradeoff between search performance and download size. For example, if we increase x to 11 bits in this example, we decrease to p.sub.f .about.1/128 at a cost of about 125K bytes in the size of the first level. Although y=0 in this example, the y bits of <partialFileCRC> 440 can be used to extend the tradeoff range as N becomes very large, or in the unusual case where many files with the same name/time/date/size (i.e., <dirInfoCRC> 439) exist with different file contents (and thus <partialFileCRC> values 440). The Agent 108 determines all these parameters at database creation time based on the statistics of the entries in the database. In the preferred embodiment, m.sub.0 +x is always at least 16, meaning that the first level entries contain at least the 16 most significant bits of the <dirInfoCRC> value 439, so that only the least significant 16 bits of <dirInfoCRC> 439 need to be kept in the second level 501 at 508. At the beginning of the backup process, the backup software of the preferred embodiment loads into memory the first level 500 of the global directory database file 145, either from the backup storage means 101 or, to minimize network bandwidth consumption, from cached copy in a directory on a disk local to the node. For each new/updated file to be backed up, a search is performed through the first level database entries to see if there is a match. If no match is found at this level (the "no match" case), there is no matching file anywhere in the database, so the backup proceeds to copy the file contents into the backup data file, which may involve computing differences from the previous file version in the case of an updated file. If a match is found at the first level, the corresponding second-level entry (or entries) is retrieved and compared; if no match is found here, the backup proceeds as in the "no match" case just discussed. The position of the corresponding second-level entry is easily determined, as discussed above, because its ordinal location in the second level is the same as the ordinal the associated first-level entry. If a match is found at the **second level, further inquiry into the backup data** file containing the <fileInfo> and <fileInfoData> records 408, 436 associated with the file may be necessary in some cases, depending on the size of the file (e.g., <fileCRC> 409 may be needed for large files) and whether the user has enabled the "exhaustive compare" mode, but in most cases a match to the global directory entry at the second level is sufficient to indicate a file match. If it is ultimately determined that a complete match has occurred, the <fileID> 214 included in the <fileEntry> record 207 of the backup directory file for this backup is set to indicate the matching file in the global database, so no file data needs to be saved in the backup data file for this backup, and there is no

new <fileIndex> 215 assigned, nor a <fileInfo> section 408 added.

US-PAT-NO: 5778395
DOCUMENT-IDENTIFIER: US 5778395 A

TITLE: System for backing up files from disk volumes on multiple nodes of a computer network

DATE-ISSUED: July 7, 1998

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Whiting; Douglas L. | Carlsbad | CA | N/A | N/A |
| Dilatush; Tom | Chula Vista | CA | N/A | N/A |

US-CL-CURRENT: 707/204; 707/10

ABSTRACT:

A system for backing up files from disk volumes on multiple nodes of a computer network to a common random-access backup storage means. As part of the backup process, duplicate files (or portions of files) may be identified across nodes, so that only a single copy of the contents of the duplicate files (or **portions thereof) is stored in the backup** storage means. For each backup operation after the initial backup on a particular volume, only those files which have changed since the previous backup are actually read from the volume and stored on the backup storage means. In addition, differences between a file and its version in the previous backup may be computed so that only the changes to the file need to be written on the backup storage means. All of these enhancements significantly reduce both the amount of storage and the amount of network bandwidth required for performing the backup. Even when the **backup data** is stored on a shared-file server, data privacy can be maintained by encrypting each file using a key generated from a fingerprint of the file contents, so that only users who have a copy of the file are able to produce the encryption key and access the file contents. To view or restore files from a backup, a user may mount the backup set as a disk volume with a directory structure identical to that of the entire original disk volume at the time of the backup.

24 Claims, 14 Drawing figures

Exemplary Claim Number: 1

Number of Drawing Sheets: 11

---------- KWIC ---------

Abstract Text - ABTX:

A system for backing up files from disk volumes on multiple nodes of a computer network to a common random-access backup storage means. As part of the backup process, duplicate files (or portions of files) may be identified across nodes, so that only a single copy of the contents of the duplicate files (or **portions thereof) is stored in the backup** storage means. For each backup operation after the initial backup on a particular volume, only those files which have changed since the previous backup are actually read from the volume and stored on the backup storage means. In addition, differences between a file and its version in the previous backup may be computed so that only the changes to the file need to be written on the backup storage means. All of these enhancements

significantly reduce both the amount of storage and the amount of network bandwidth required for performing the backup. Even when the **backup data** is stored on a shared-file server, data privacy can be maintained by encrypting each file using a key generated from a fingerprint of the file contents, so that only users who have a copy of the file are able to produce the encryption key and access the file contents. To view or restore files from a backup, a user may mount the backup set as a disk volume with a directory structure identical to that of the entire original disk volume at the time of the backup.

Detailed Description Text - DETX:

In yet another embodiment, the backup storage means incorporates hierarchical storage management (HSM), in which files that have not been accessed for a long time are migrated from disk to a secondary storage means, such as tape or optical disk. The main purpose of HSM is to save on storage costs for very large storage systems by providing the management tools that allow the migration to be transparent to the system, except for the additional delay in accessing some files. Use of any form of HSM in conjunction with the backup storage means of the present invention does not significantly affect any of the concepts discussed here. However, care must be taken not to impair performance of the backup and restore operations, since delays incurred in accessing secondary storage may render the system much less usable. Indeed, it would be fairly simple to identify **portions of the contents of backup data** and directory files of the present invention which could be migrated to secondary storage without adversely affecting backup performance. Fortunately, in most cases, the data reduction methods of the present invention are sufficiently powerful to keep disk storage costs down to an acceptable level even without using HSM.

Detailed Description Text - DETX:

The backup process of the preferred embodiment actually creates two files containing information about each backup set: a backup directory file (e.g., 143), and a **backup data** file (e.g., 144). In an alternate embodiment, these files could be combined into a single file. The contents of the backup directory file indicate the directory structure of the source disk volume, as well as pointers into the **backup data** files (e.g., 144, 149, and **backup data** files of other users) indicating where the data for each file is to be found. One key feature of the present invention is the data reduction achieved by duplicating pointers to data and directory information instead of duplicating the information itself, including referencing duplicate information across users. To explain the role of the backup directory file(s) in accomplishing this data reduction in the preferred embodiment, a description of key **portions of the backup** directory file (e.g., 143) for a DOS disk volume is given in FIG. 4 in Backus-Naur Form (BNF), which is a well known formal language technique (for example, see Nicklaus Wirth, Algorithms+Data Structures=Programs, 1976, pp. 281-291). Before discussing the contents of FIG. 4, we will explicitly define the conventions of our BNF, since there are slight variations in syntax from one author to the next. Non-terminals are enclosed in angle brackets (e.g., <fileEntry>) . The :==symbol indicates a formal definition. Terminals are indicated as single binary digits (0 or 1), or as hexadecimal quantities using C-like syntax: 0xUU for 8-bit bytes, 0xUUUU for 16-bit words, and 0xUUUUU for 32-bit dwords. Ranges of terminal values are indicated as two terminal quantities with two periods in between; e.g., 0.times.00 . . . 0.times.FE. The .vertline. character is a meta-symbol indicating "one or the other", while brackets [ ] indicate an optional field, and an asterisk (*) indicates one or more repetitions of the field. Thus, for example, [<externDirItem>]* indicates zero or more of the non-terminal <externDirItem>. The double slash // indicates a comment to the end of the line.

US-PAT-NO: 6473794
DOCUMENT-IDENTIFIER: US 6473794 B1

TITLE: System for establishing plan to test components of web based framework by displaying pictorial representation and conveying indicia coded components of existing network framework

DATE-ISSUED: October 29, 2002

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
| --- | --- | --- | --- | --- |
| Guheen; Michael F. | Tiburon | CA | N/A | N/A |
| Mitchell; James D. | Manhattan Beach | CA | N/A | N/A |
| Barrese; James J. | San Jose | CA | N/A | N/A |

US-CL-CURRENT: 709/223; 709/224

ABSTRACT:

A system, method, and article of manufacture are provided for planning the testing of components of an existing network framework. First, a pictorial representation of an existing network framework is displayed along with a plurality of components of the existing network framework. Thereafter, the components of the existing network framework are indicia coded in order to convey a plan by which the components of the existing network framework are to be tested. The components may be indicia coded in order to convey an order of the testing or which components of the existing network framework are to be tested.

19 Claims, 177 Drawing figures

Exemplary Claim Number: 1

Number of Drawing Sheets: 177

---------- KWIC ---------

Detailed Description Text - DETX:

What Other Utilities are Available With the Tool? The backup product should have fundamental management capabilities. Automatic restore, unattended operation and command line processing of the product should be available. Basic tape functions such as cataloging, internal labeling, initialization, certification, scratch protection and write protection are musts. Performs automatic **backup of data** files on site standards. Designed along the lines requester-server model; more specifically the tool runs on the server machine and acts as a shared resource for data access, integrity, security recovery, etc. Full auditing capability should be present for backups as well as error detection and notification that a backup has failed should be available. Provide full and incremental **backups, partial** restore, and compression/decompression. Capable of managed and systematic restore process.

US-PAT-NO: 6353878
DOCUMENT-IDENTIFIER: US 6353878 B1

TITLE: Remote control of backup media in a secondary storage subsystem through access to a primary storage subsystem

DATE-ISSUED: March 5, 2002

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Dunham; Scott R. | Billerica | MA | N/A | N/A |

US-CL-CURRENT: 711/162; 707/10 ; 707/204 ; 707/9 ; 711/112 ; 711/114 ; 711/161 ; 711/4

ABSTRACT:

A primary data storage subsystem has primary data storage and a storage controller for controlling access of host processors to the primary data storage. The primary data storage subsystem is linked to a secondary data storage subsystem for transfer of backup data between the primary data storage subsystem and the secondary data storage subsystem. The secondary data storage subsystem has a tape library unit for storing the backup data. A host processor usually accesses the primary storage. A host can also send backup and restore commands to the storage controller, to cause specified data from the primary storage to be written as a backup version in the tape library unit, and to cause a specified backup version to be read from the tape library unit to be accessible to the host from the primary data storage unit. In this fashion, the host need not be concerned with the basic operations of the tape library unit. A host, however, can send backup media remote control requests to the storage controller, and the storage controller responds by sending corresponding backup media remote control commands to the tape library unit, to permit the host to control remotely the basic operations of the tape library unit. Therefore, the host can obtain status of read/write stations in the tape library unit, and control the mounting, unmounting, and transport of the tapes mounted at the read/write stations.

30 Claims, 17 Drawing figures

Exemplary Claim Number: 1

Number of Drawing Sheets: 16

---------- KWIC ---------

Detailed Description Text - DETX:

The data mover computers 72-75 and the control station servers 76-77 are commodity personal computers. The data mover computers 74, 75 provide a front-end for the secondary data storage subsystem 43, and they are programmed to respond to backup and restore commands from the primary data storage subsystems. In response to a backup request, a front-end data mover computer 74, 75 moves the **backup data** to the cached disk data storage subsystem 71, updates the secondary directory 48, and initiates the transfer of the **backup data** from the cached disk data storage subsystem 71 to the tape library unit 70. The actual transfer of the **backup data** from the cached disk data storage subsystem 71 to the tape library unit 70 is performed by one of the back-end

data mover computers 72, 73. In response to a restore request, a front-end data mover computer 74, 75 accesses the secondary directory 48 to determine the most accessible source of the **backup data** (cache memory 86, disk array 87, or tape cassettes 85), and accesses the **backup data** from the cache memory 86 or the disk array 87, or if the **backup data** is not accessible from the cache memory 86 or the disk array 87, the front end data mover sends a command over the 10-Base-T bus 79 to one of the back-end data mover computers 72, 73 to read the **backup data** from the tape cassettes and transfer the data from the tape cassettes to the cache memory 86 in the cached disk data storage subsystem 71. Once at least a **portion of the backup data** has been transferred from tape 85 to the cache 86, the front-end data mover computer 74, 75 transfers the **backup data** from the cache memory 86 to the primary data storage subsystem having issued the restore request.